

# DUBLIN CITY UNIVERSITY

## SEMESTER ONE EXAMINATIONS 2007

**MODULE:** Compiler Construction 1  
**(Title & Code)** CA448/CA423

**COURSE:** B.Sc. in Computer Applications (SE)  
B.Sc. in Computer Applications (CSSE)  
B.Sc. in Computer Applications (IS)  
B.Sc. in Computational Linguistics

**YEAR:** 4

**EXAMINERS:** Dr. F. Bannister  
Dr. P. Gibson  
Dr. G. Hamilton (ext. 5017)

**TIME ALLOWED:** 2 Hours

**INSTRUCTIONS:** Please answer ALL questions.  
All questions carry equal marks

*Requirements for this paper*  
*Please tick (X) as appropriate*

<input type="checkbox"/>	<i>Log Table</i>
<input type="checkbox"/>	<i>Graph Paper</i>
<input type="checkbox"/>	<i>Attached Answer Sheet</i>
<input type="checkbox"/>	<i>Statistical Tables</i>
<input type="checkbox"/>	<i>Floppy Disk</i>
<input type="checkbox"/>	<i>Actuarial Tables</i>

**THE USE OF PROGRAMMABLE OR TEXT STORING  
CALCULATORS IS EXPRESSLY FORBIDDEN**

**PLEASE DO NOT TURN OVER THIS PAGE UNTIL YOU ARE  
INSTRUCTED TO DO SO**

1. Write a regular expression which describes the language of binary numbers from the alphabet  $\{0,1\}$  which are either odd or a power of 2 (or both). [10 marks]

2. Construct a non-deterministic finite state automaton (NFA) that recognises the same language as defined by the regular expression in question 1. [10 marks]

3. Prove that the following grammar is ambiguous: [10 marks]

$$E \rightarrow E + E \mid E * E \mid \sim E \mid (E) \mid \text{num}$$

by giving two different parse trees for the following expression:

$$4^*(\sim 3+5)$$

4. Construct an unambiguous grammar which describes the same language as the grammar in question 3, and which deals with the precedence of the arithmetic operators (unary minus ( $\sim$ ) has the highest precedence, followed by multiplication ( $*$ ), followed by addition ( $+$ )). [10 marks]

5. Verify whether or not the following grammar is LL(1): [10 marks]

$$\begin{aligned} S &\rightarrow ABa \\ A &\rightarrow bA \\ A &\rightarrow \varepsilon \\ B &\rightarrow aB \\ B &\rightarrow \varepsilon \end{aligned}$$

6. Verify whether or not the following grammar is LR(0): [10 marks]

$$\begin{aligned} S &\rightarrow E \\ E &\rightarrow T \\ E &\rightarrow E;T \\ T &\rightarrow \varepsilon \\ T &\rightarrow Ta \end{aligned}$$

7. Verify whether or not the grammar given in question 6 is SLR(1). [10 marks]

8. Verify whether or not the following grammar is LR(1): [10 marks]

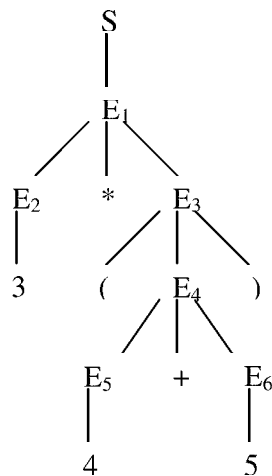
$$\begin{aligned} S &\rightarrow A \\ A &\rightarrow bAb \\ A &\rightarrow \varepsilon \end{aligned}$$

9. Consider the following attribute grammar:

[10 marks]

Production	Semantic Rules
$S \rightarrow E$	$E.place := newtemp$
$E \rightarrow E_1 + E_2$	$E.value := E_1.value + E_2.value$ $E_1.place := E.place$ $E_2.place := newtemp$
$E \rightarrow E_1 * E_2$	$E.value := E_1.value * E_2.value$ $E_1.place := E.place$ $E_2.place := newtemp$
$E \rightarrow (E_1)$	$E.value := E_1.value$ $E_1.place := E.place$
$E \rightarrow num$	$E.value := num$

Give a possible order of evaluation of these attributes for the following expression tree:



10. Convert the following statement into three-address code.

[10 marks]

$$x := ((b - d) * 4 + a) * c / (2 * e)$$

[Total marks: 100]